CO3102/CO7102

Coursework 2 – Mini Web/App Project

SLPP – Shangri-La Petition Platform

Important Dates:

Deadline: 13th -Jan-2025 at 11:59 GMT

Please ensure that you submit your work on time.

- This coursework counts as 25% of your final mark.
- Please read guidelines on plagiarism on
- https://le.ac.uk/policies/regulations/senate-regulations/senate-regulation-11/academic
  Learning outcome:
  - Use appropriate server-side and client-side scripting languages to create a web application.
  - Solve security and session handling issues and use supporting techniques.

Coursework Description

Citizens in the Valley of Shangri-La now have a new way to shape parliamentary discussion through the **Shangri-La Petition Platform (SLPP).** This platform allows citizens of Shangri-La to create or sign petitions on matter within the government's responsibility. Any citizen can propose a petition on topics they care about. Once a petition reaches the signature threshold set by the Petitions Committee, it then qualifies for parliamentary debate. After the debate, the Petitions Committee issues a summary response of behalf of the Parliament before formally closing the petition. Your task is to design and implement the SLPP, which can be developed as either an app or a website.

Requirements

You task is to develop the Shangri-La Petition Platform (SLPP), designed to give Shangri-La citizens a voice. SLPP can be implemented as a web application, a native Android/iOS app, or a hybrid app.

SLPP should also provide open access to the petition data and statistics through its Open Data API. The general public, media, and organisations will be able to access petition records using a REST API.

Task 1 – Website or Mobile App (80%)

Your first task is to implement SLPP as w website or mobile app, including the front-end and backend.

There are two types of accounts in SLPP (1) **Petitioner** and (2) **Petitions Committee**

(1) Petitioner

A petitioner must first register to be able to create or sign a petition. During registration, petitioners are required to provide the following information:

- Email (as login id)
- Full Name
- Date of Birth
- Password
- Biometric ID (BioID) – a unique 10-digit code assigned to Shangri-La citizens.
  *Refer to Appendix 2 for more details.

Every Shangri-La citizen has been given a Biometric ID (BioID), which can be found on their birth or naturalisation certificates, and the corresponding QR code of BioID is also printed on the certificates. To complete the registration process, a petitioner should either enter a valid BioID or scan the QR (refer to Appendix 2). Once the petitioner account is created, they can sign in and access the **Petitioner Dashboard,** where a petitioner can:

- Create a new petition (with title and content)
- View a list of all petitions and their details, status (open/closed) and response, if any.
- Sign any open petition proposed by others

Note that

- Only logged-in petitioners can access the Petitioner Dashboard.
- A petitioner may create multiple petitions and sign any petition proposed by others.
- Once a petition is created, it will be open to signatures, and it cannot be closed, deleted or edited by the petitioner.
- Similarly, once a petition is signed, the signature cannot be revoked.
- A petitioner can only sign a specific petition once.

Please ensure that all Petitioners' passwords are stored securely.

(2) **Petitions Committee**

There is one pre-defined Petitions Committee account with the login name "admin@petition.parliament.sr" and a default password "2025%shangrila". The Petitions Committee Officer has access to the Petitions Committee Dashboard to:

- Set signature thresholds that apply to all petitions.
- View details for all petitions, including their status and the total number of signatures gathered.
- Write a brief response on behalf of the Shangri-La Parliament for any petition that meets the required signature threshold. Once a response is entered, the petition’s status will be updated to "closed" (no longer accepting new signatures).

Error handling

SLPP should display appropriate messages (either an error page or an Ajax message) in the following situations:

- When the BioID entered does not match the record in the database during registration.
- When the BioID has already been used by another user or the QR code has already been scanned.
- When the provided email address is already associated with an existing registered petitioner.
- When an invalid username or password is entered.

Task 2 - REST Web Service(20%)

Your second task is to implement “SLPP Open Data REST API” according to the specification below:

2.1 return the details of all petitions

HTTP request:

```
GET /slpp/petitions
```

An example of JSON Response:

```
{
    "petitions": [
        {
            "petition_id": "1",
            "status": "closed",
            "petition_title": "Increase Funding for Primary Care in SLHS in the 2025 
Budget",
            "petition_text": "The Shangri-La government should consider increasing the 
funds allocated to primary care in the 2025 budget for the Shangri-La Health Service 
(SLHS)...",
            "petitioner": "johnsmith@gmail.com",
            "signatures": "999",
            "response": "Parliamentary debates Note: the government will consider the 
proposal if..."
        },
        {
            "petition_id": "2",
            "status": "open",
            "petition_title": "Reduced or subsidized train fares for University 
students",
            "petition_text": "A University student-specific discount (80% off) on train 
tickets to make transportation more affordable..",
            "petitioner": "janeroe@gmail.com",
            "signatures": "3",
            "response": "Parliamentary debates Note: the government will consider the 
proposal if..."
        }
    ]
}
```

2.2 Return the details of all open petitions

```
GET /slpp/petitions?status=open
```

An example of JSON response

```
{
    "petitions": [
        {
            "petition_id": "2",
            "status": "open",
            "petition_title": "Reduced or subsidized train fares for University 
students",
            "petition_text": "A University student-specific discount (80% off) on train 
tickets to make transportation more affordable..",
            "petitioner": "janeroe@gmail.com",
            "signatures": "3",
            "response": "Parliamentary debates Note: the government will consider the 
proposal if..."
        }
    ]
}
```

Marks breakdown

(1) Petitioner registration, log-in/sign-out.	[30 marks]
(2) Petitioner Dashboard	[30 marks]
(3) Petitioner Committee Dashboard	[20 marks]
(4) REST API	[20 marks]

Note that

- For solutions to (1), (2), and (3), you may choose to develop either a web application, a native mobile app (Android or iOS), or a hybrid app. You are free to use any programming languages or frameworks. Please refer to Appendix 1 for further details.
- Use appropriate techniques to remember the username from the last login username such as using cookies or shared preferences.

You're welcome to use the **Petition.sql** file provided on Blackboard for this coursework, and you're free to make any changes if necessary. However, you are **NOT** required to use this file, especially if you plan to use NoSQL databases (e.g., MongoDB, Firebase, etc.) or other data persistence frameworks (e.g., Spring JPA). If you intend to use the departmental MySQL server (mysql.mcscw3.le.ac.uk) for this coursework, please ensure that you test the connection string before submitting your work

Submission

- Compress all files in a single zip file for submission via Blackboard:
  - Your Project folder
  - READ.ME (explaining how to deploy and run you application)
  - Your database schema and all data, if applicable (Your_email_ID.sql)
- The archive should be named **CW2_abc123.zip**. Where abc123 is your email id (e.g. CW2_yh37.zip)

You are allowed to re-submit as many times as you like **before** the deadline.

Anonymous marking

We operate an anonymous marking scheme. All submitted submission (or other project files) will be deployed anonymous using the fingerprinting generated by SHA256.

Appendix

(1) Languages & frameworks

You may use **ANY** programming languages for this coursework, including but not limited to:

- Java - Android Studio
- JavaScript -React Native, Ionic etc
- PHP
- Python
- .NET

- Swift/Objective-C – Xcode - iOS
- Any other languages/web technologies for developing Progressive Web Apps

You may use **ANY** framework for this part, including but not limited to:

- Spring MVC
- ASP .NET MVC
- Ruby On Rails
- Node.js/React.js
- Laravel
- Flask
- Angular
- Django
- Ember.js

You are allowed to use any Third-party CSS/JS/HTML libraries e.g., Bootstrap. The architecture and good coding practice will also be considered when allocating marks; For Native app, you are allowed to build you application based on a template. Please email yh37@leicester.ac.uk if you are unsure whether your chosen framework is permitted.

(2) Valid BioID code and QR codes

Below is a list of all valid Shangri-la BioID. Petitioners are required to enter one of the provided codes to complete the registration process.

K1YL8VA2HG	V30EPKZQI2	QJXQOUPTH9	CET8NUAE09	BZW5WWDMUY
7DMPYAZAP2	O3WJFGR5WE	GOYWJVDA8A	VQKBGSE3EA	340B1EOCMG
D05HPPQNJ4	SEIQTS1H16	6EBQ28A62V	E7D6YUPQ6J	CG1I9SABLL
2WYIM3QCK9	X16V7LFHR2	30MY51J1CJ	BPX8O0YB5L	49YFTUA96K
DHKFIYHMAZ	TLFDFY7RDG	FH6260T08H	AT66BX2FXM	V2JX0IC633
LZK7P0X0LQ	PGPVG5RF42	JHDCXB62SA	1PUQV970LA	C7IFP4VWIL
H5C98XCENC	FPALKDEL5T	O0V55ENOT0	CCU1D7QXDT	RYU8VSS4N5
6X6I6TSUFG	2BIB99Z54V	F3ATSRR5DQ	TTK74SYYAN	S22A588D75
QTLCWUS8NB	ABQYUQCQS2	1K3JTWHA05	4HTOAI9YKO	88V3GKIVSF
Y4FC3F9ZGS	9JSXWO4LGH	FINNMWJY0G	PD6XPNB80J	8OLYIE2FRC

Example of QR codes:

K1YL8VA2HG

QR codes of all valid BioID are available to download from Blackboard

Feel free to edit the sample records in **Petition.sql** or make any changes you deem necessary.

(3) Miscellaneous

Default Admin credentials
"admin@petition.parliament.sr" and a default password "2025%shangrila".

```
     public static String getSHA256(String data) {
         String result = null;
         try {
             MessageDigest digest = MessageDigest.getInstance("SHA-256");
             byte[] hash = digest.digest(data.getBytes("UTF-8"));
             return bytesToHex(hash); // make it printable
         }catch(Exception ex) {
             ex.printStackTrace();
         }
         return result;
     }
     private static String bytesToHex2(byte[] hash) {
        // return DatatypeConverter.printHexBinary(hash);
        final StringBuilder builder=new StringBuilder();
        for(byte b:hash) {
                builder.append(String.format("%02x", b));
        }
        return builder.toString();

     }
…
```

Marking Scheme

Task 1

N (0%)

- No submission

F (0-40%)

- Static pages (or View/Activity) for sign-up/login exists, but the petitioners are unable to register or log in.
- No user authentication. Not roles/permissions control.
- Petitioner and Petition Committee Dashboard not implemented.

E (40-50%)

- Web app/Hybrid app: Basic HTML pages/view without any front-end framework or CSS; do not adopt a responsive design; no form validation; major issue with sign-up and login page.
- Native or Hybrid App: basic Page/Activity layouts exist but without any backend; backend issues with the; no form validation; major issue with sign-up and login page.
- Petitioner registration page partially implemented (e.g., email/full name etc。) without BioID validation
- Petitions Committee page exists, but with hard-coded data.

D (50-60%)

- The adoption of responsive web design for Web App; Native or Hybrid App
- Server-side BioID validation implemented.
- Sign-up page and login page works to a certain extent. User authentication and DB backend partially implemented, though there may still be major issues. (e.g., failed to connect, 500 internal server error)
- Petition signing page exists and partially functional.
- Petitions Committee page exists and password-protected, though there are issues with the results.

C (60-70%) Meeting the criteria above in D, and

- Sign-up and Login pages work reasonably well; necessary form validation implemented.
- A petitioner can enter the personal detail and manually enter BioID to complete the registration.
- Petitioner can view and petitions available petitions.
- Petitions Committee dashboard fully functional.

B (70-85%) Meeting the criteria above in C, and

- Sign-up and Login pages work very well.
- User-friendly sign-up form validation with detailed error messages (e.g. Use of Ajax for form validations and BioID verification)
- A petitioner can scan the QR code to autofill BioID.
- The Petitions Committee dashboard fully functional; results being correctly displayed.
- Use Cookies or Shared Preference to remember last login name.
- Use appropriate chart to visualise the result in the Petitions Committee dashboard (e.g. Google Chart, C3 JS or Chart JS etc)
-

A (>85%) Meeting the criteria above in B, and

- Secure RESTful Service such as OAuth2/JWT Tokens.
- Data mashup in Petitions Committee dashboard such as topic word cloud etc.

Bonus:

- Extra bonus points will be awarded for additional features, included but not limited to: (Please complete "CW2 Self-assessment form" attached.

Task 2

The testing process for the REST Service API will be automated. A set of test cases will be executed for each REST service endpoint to verify that they function as expected. Marks will be allocated based on the results of these tests..

How your score will be calculated:

$$S_{overall} = \frac{\sum_{i=1}^{p} w_i}{\sum_{i=1}^{n} w_i} \times 80\,\% + S_{part2} \times \ 20\%$$

(Where *p* is the number of passed test cases, *n* is the total number of test cases, *wi* refers to the weight for each test case)